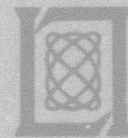②

Semiannual Technical Summary

Distributed Sensor Networks

AD–A160 596

DTIC
S ELECTE D
OCT 2 2 1985
B

26 SEPT. 85
~~31 March 1985~~

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS

85 10 11 033

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

DISTRIBUTED SENSOR NETWORKS

SEMIANNUAL TECHNICAL SUMMARY REPORT
TO THE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

1 OCTOBER 1984 — 31 MARCH 1985

26 SEPTEMBER 1985

DTIC
S ELECTE D
OCT 2 2 1985
B

LEXINGTON   MASSACHUSETTS

ABSTRACT

This report describes the work performed on the DARPA
Distributed Sensor Networks Program at Lincoln Laboratory
during the period 1 October 1984 through 31 March 1985.

iii

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

DISTRIBUTED SENSOR NETWORKS

I. INTRODUCTION AND SUMMARY

The Distributed Sensor Networks (DSN) program is aimed at developing and extending target surveillance and tracking technology in systems that employ multiple spatially distributed sensors and processing resources. Such a system would be made up of sensors, data bases, and processors distributed throughout an area and interconnected by an appropriate digital data communication system. The detection, tracking, and classification of low flying aircraft has been selected to develop and evaluate DSN concepts in the light of a specific system problem. A DSN test-bed has been developed and is being used to test and demonstrate DSN techniques and technology. The overall concept calls for a mix of sensor types. The initial test-bed sensors are small arrays of microphones at each node augmented by TV sensors at some nodes. This Semiannual Technical Summary (SATS) reports results for the period 1 October 1984 through 31 March 1985.

Progress in the development of distributed tracking algorithms and their implementation in the DSN test-bed system is reviewed in Section II. Test-bed versions of distributed acoustic tracking algorithms now have been implemented and tested using simulated acoustic data. This required developing a solution to a basic distributed tracking problem: the information feedback problem. Target tracks received by one node from another node often implicitly include information that originally was obtained from the receiving node. The receiving node must take account of this correctly when combining the received track with its own tracks, or unstable information feedback will occur. This did occur with an initial version of our distributed tracking algorithms. A symbolic simulation of the situation was implemented, the source of the problem identified, and a solution developed. The algorithms now require each node to maintain two tracks for each target, a local track and a common track, and to broadcast both so that receiving nodes can incorporate the received tracks with their own without producing any information instability. Additional test-bed tracking software modifications now are being implemented to incorporate azimuth measurements derived from TV sensors and to provide additional experimental convenience. A joint algorithm research effort has been undertaken with Advanced Information and Decision Systems (AI&DS) to investigate the application of their multihypothesis tracking approach.

An acoustic tracking algorithm evaluation plan has been formulated. Initial algorithm shakedown and tuning experiments have been completed using prerecorded acoustic data and simulated data. Section III reviews the plan and initial results.

Section IV summarizes the results of some initial research into distributed classification methodology. The emphasis is upon distributed implementations of and approximations to optimal centralized statistical classification methods.

A test-bed TV subsystem is being developed for experimentation with TV-aided acoustic tracking. Section V reviews progress with the subsystem hardware as well as with algorithms and software for target selection, camera control, image processing, and target detection.

Section VI reviews research progress towards the implementation of an acoustic signal processor diagnosis system. The system makes use of some of the latest A.I. techniques such as qualitative reasoning and hierarchical planning. The system performs diagnosis on the basis of abstract Fourier domain models of signal processing rather than on the basis of the implementation details of algorithms that necessarily must operate upon time domain signals.

The final section treats a number of smaller items including conversion of software between computer systems, test-bed maintenance and operational experience, addition of nodes to the test-bed and development of communication software.

2

II. DISTRIBUTED TRACKING ALGORITHM DEVELOPMENT

The implementation and validation of real-time distributed aeroacoustic tracking algorithms for the DSN test-bed were completed during this reporting period. An initial version was completed early in the reporting period and tested in a non-real-time VAX-11/780 environment. This version was unstable for certain situations with more than two nodes and frequent internodal broadcasts. A computer program to perform a symbolic simulation of information flow between nodes was developed in the MACSYMA language and used to find a solution to the instability problem. The tracking algorithm design was modified and the modified algorithm was implemented in the test-bed. It is now in routine use for tracking experiments. Late in the reporting interval, we began additional modifications to the tracking software to make use of azimuth measurements from a TV subsystem and to improve the experimental convenience and flexibility of the test-bed. We also have initiated a joint algorithm development effort with Advanced Information and Decision Systems (AI&DS). The following provides more technical details.

A. BROADCAST COMMUNICATIONS AND INFORMATION FLOW

DSN nodes broadcast tracking information and use information received from other nodes. This raises two distinct distributed tracking questions: (1) when to broadcast track information, and (2) what information to broadcast and how to use it. What to broadcast and how to use it were central research questions in the development of the distributed tracking algorithms that are now operational in the test-bed. The remainder of this section summarizes the results of this research in terms of the development of the now operational distributed tracking algorithms.

The distributed tracking algorithms require each node to maintain a pair of tracks for each target. One of these, the 'local' track, is based on all information known to the node, including its own recent sensor measurements that have been made since the last broadcast of information to other nodes. The other track, the 'common' track, utilizes only a subset of this information: the information that has been received from or broadcast to other nodes. The common track is based only upon shared information while the local track includes information that is only available locally. Each track, common or local, consists of target state vectors and covariance matrix estimates.

We initially elected to broadcast only local tracks. The algorithm for updating local and common tracks implicitly assumed that broadcast messages would not be delayed or lost and that only one node would broadcast at a time. We further assumed that all nodes that could detect a given target simultaneously would be within broadcast range of each other. This resulted in the following simple track update algorithms.

The common track of a node is set equal to its own local track whenever the node broadcasts. Upon receipt of a broadcast, a node takes one of two actions, depending upon whether or not the received track is for a target already known to the receiver. The decision is

3
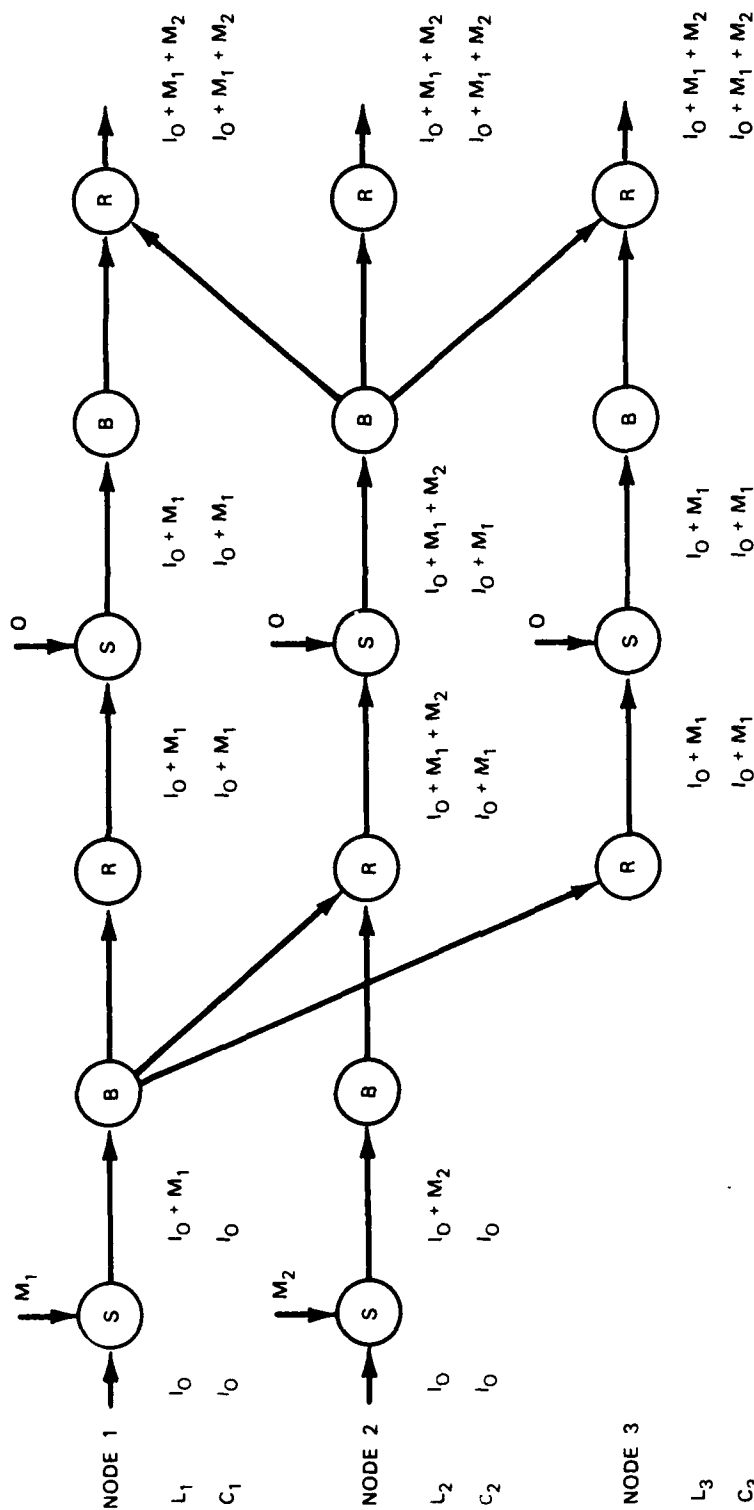
Figure II-1. Example of correct tracking information flow for target handover situations with one target, three nodes, two measurements, two local track broadcasts, and two receptions. Circles correspond to information transformations: those labelled S for sensing, B for broadcast, and R for reception. Information in each track is indicated symbolically by algebraic expressions. L_i is local track information at node i and C_i is the common track information at node i.

154890-N

made on the basis of the track identifier, which is an identifier assigned to the track when it is first initiated. If the received track identifier is new to the receiving node, then new local and common tracks are created. The new tracks are assigned the identifier, state, and covariance matrix of the received track. If the identifier of the received track matches that of an existing local and common track pair, the three tracks are combined to update the local track, and the common track is set equal to the received track.

The information in the local and received tracks is partially redundant because they contain common track information as well as information derived from their own sensors. To update a local track correctly, one must sum the information in the old local track and the received track and subtract the information in the common track. The equations for the necessary information arithmetic are:

$$\Sigma'_L \leftarrow \left[\Sigma_L^{-1} + \Sigma_R^{-1} - \Sigma_C^{-1} \right]^{-1}$$

and

$$X_L \leftarrow \Sigma'_L \left[\Sigma_L^{-1} X_L + \Sigma_R^{-1} X_R - \Sigma_C^{-1} X_C \right]$$

followed by

$$\Sigma_L \leftarrow \Sigma'_L$$

where X_L and Σ_L are the state estimate and covariance matrix of the local track, respectively, X_R and Σ_R are the state estimate and covariance matrix of the received track, and X_C and Σ_C of the common track.

Figure II-1 shows a symbolic illustration of the correct behavior of this algorithm for the specific case of one target, three nodes, two measurements, two broadcasts, and two receptions. Nodes 1 and 2 start with identical local and common tracks, and node 3 has no information about the target. In the first measurement cycle, nodes 1 and 2 detect the target and separately increase the information in their local tracks. Then node 1 broadcasts and both other nodes receive the broadcast. After reception, all three nodes have identical common tracks. Node 2 has a different local track since it has not yet shared its measurement information. Node 3 now has a local and common track pair although it has not yet detected the target. In the second measurement cycle, no node detects the target and no information is added to any local track. Then node 2 broadcasts, and subsequently all local and common tracks become identical.

It was noted above that the algorithm implicitly assumes every broadcast about a target is received by every node that can detect the target. When this is not true, different versions of the common track will exist at different nodes, but the distributed algorithm will not fail. Figure II-2 shows another information flow diagram that demonstrates the effect of lost messages. Suppose that node 1 broadcasts a local track and node 3 receives the broadcast but node 2 does not. The figure shows that information from the first measurement is lost permanently. This is the worst that can happen and is tolerable if it occurs infrequently.

The algorithm is similarly tolerant of simultaneous broadcasts (more properly, simultaneous receptions) if all but the first of several simultaneous messages are discarded. The discarded
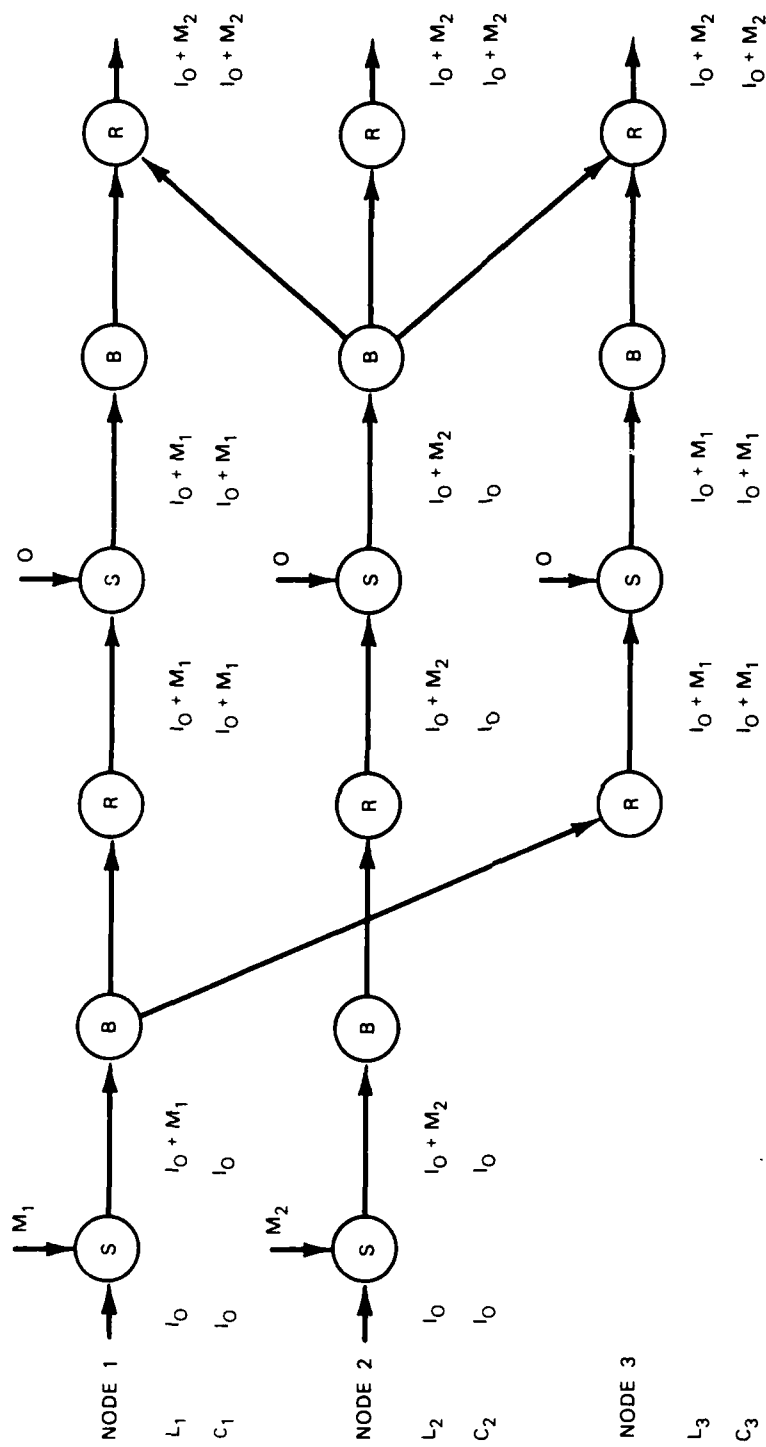
5

Figure II-2. Example of information flow and distributed tracking system recovery from a lost message even when only local tracks are broadcast. See Figure II-1 for symbology.

154891-N

messages and their associated information are lost to the system. But this is not a satisfactory solution when nodes transmit after each measurement, resulting in many simultaneous and therefore discarded messages.

Our first serious attempt at handling simultaneous broadcast was a simple extension of the algorithm described above. When multiple messages about a known target are received simultaneously, they can be processed in a batch. Information arithmetic can be used to add the information in the received tracks to the local track information. The information in the common track then can be subtracted, once for each foreign track, to cancel the overlapping information. Similar information arithmetic then can be used to update the common track. The successful operation of this extension is illustrated in Figure II-3 for the case of three nodes that all initially know about the target.

Unfortunately, when multiple messages are received simultaneously about an unknown target, the receiving node has no common track available for performing the information arithmetic required to combine the tracks. One of several alternatives is to initialize the local and common tracks with one of the received messages and discard all the others. All alternatives can result in different common tracks in different nodes. We hoped the differences would be transient, as they are for the case of nonsimultaneous broadcasts and lost messages, but this is not the case, as Figure II-4 illustrates. Starting immediately after the second reception, tracks contain extra positive and negative copies of some of the information in the system. The number of extra copies (positive and negative) increases geometrically. The result is oscillating and growing state estimation errors in local and common tracks, which we observed experimentally with our first implementation of distributed tracking algorithms before we fully understood the nature of the information stability problem and identified a solution.

Information flow charts such as those shown in Figures II-1 through II-4 were important tools that we used to understand the information stability problem and formulate a solution. Computer programs were written to trace numerically and symbolically the flow of information in the distributed algorithm. Initially, we implemented a program to trace the numeric values of state and covariance matrices. We subsequently implemented a much more useful program to trace information flow in the symbolic form shown in Figures II-1 through II-4. The numeric program verified that we had a problem, but was not very helpful in explaining its fundamental nature. The second program made the nature of the problem immediately obvious. Moreover, it revealed other potential information feedback problems with algorithms that broadcast only local tracks. This is one of the first examples of using symbolic processing to analyze information flow in a distributed system, and it highlights the potential value of symbolic rather than numeric simulation.

With the help of the information flow diagram generator, a solution to the information flow problem was quickly identified. The solution is to broadcast local and common track pairs instead of just local tracks. When it is necessary to cancel overlapping information in a received track, the accompanying received common track is used in favor of the receiving node's common track. This results in more communications traffic, but is necessary because local tracks alone are not sufficient to avoid information feedback problems.
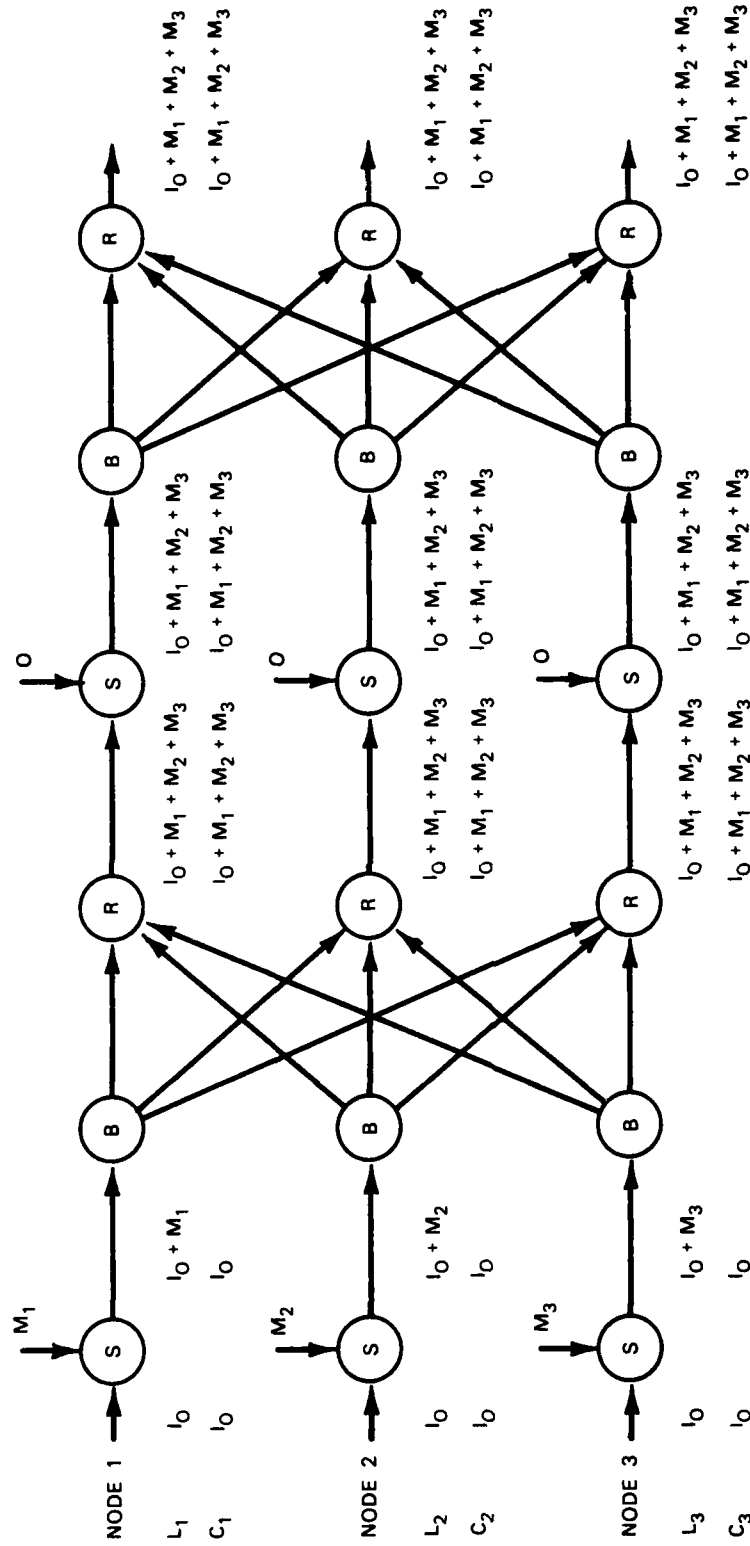
Figure II-3. Example of stable information flow for the case of multiple simultaneous transmissions by three nodes for a single known target. See Figure II-1 for symbology.

154892-N

8

NODE 1

L_1
C_1

M_1

I_0 $I_0 + M_1$
I_0 I_0

$I_0 + M_1 + M_2$
$I_0 + M_1 + M_2$

O

$I_0 + M_1 + M_2$
$I_0 + M_1 + M_2$

$I_0 + M_2$
$I_0 + M_2$

NODE 2

L_2
C_2

M_2

I_0 $I_0 + M_2$
I_0 I_0

$I_0 + M_1 + M_2$
$I_0 + M_1 + M_2$

O

$I_0 + M_1 + M_2$
$I_0 + M_1 + M_2$

$I_0 + M_2$
$I_0 + M_2$

NODE 3

L_3
C_3

$I_0 + M_2$
$I_0 + M_2$

O

$I_0 + M_2$
$I_0 + M_2$

$I_0 + 2M_1 + M_2$
$I_0 + 2M_1 + M_2$
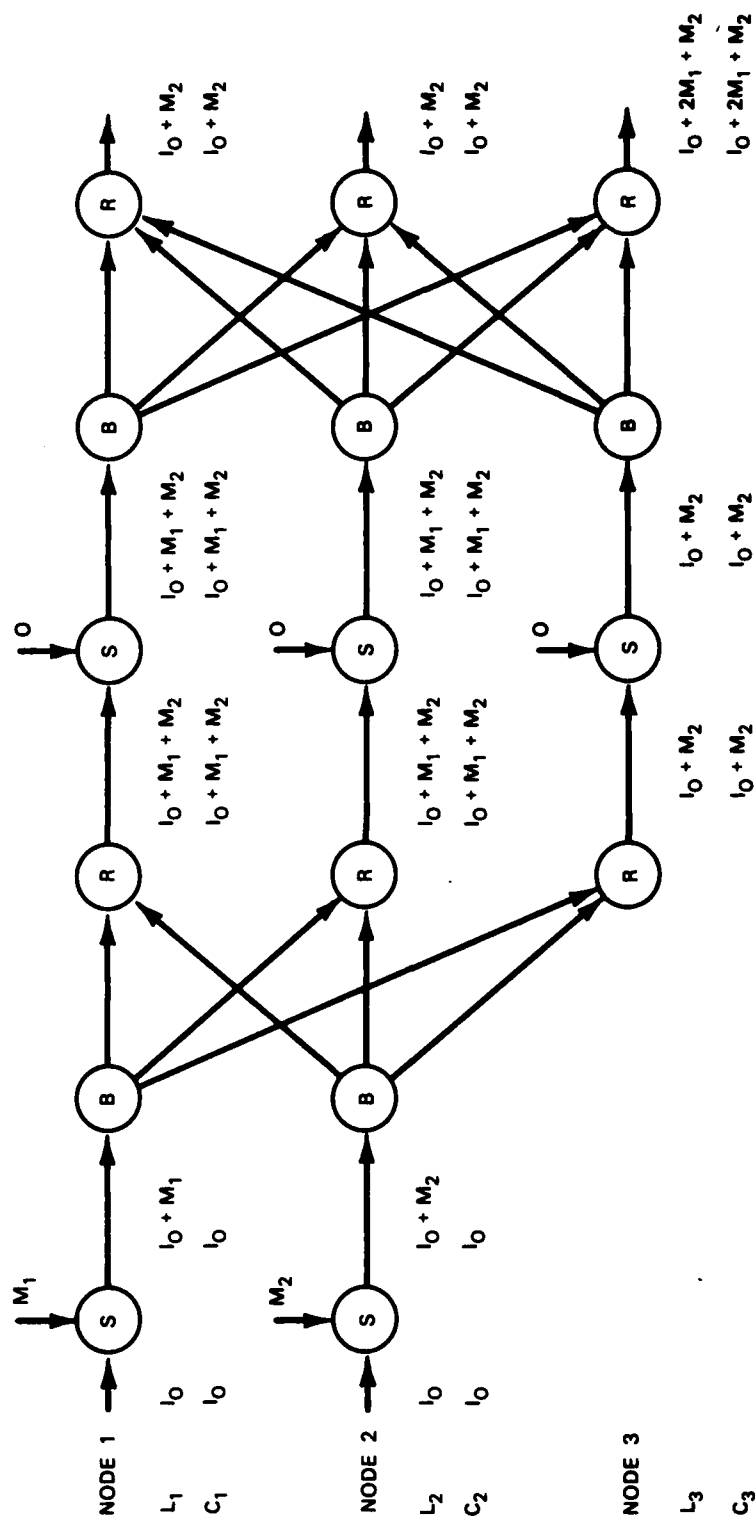
Figure II-4. Example of divergent information flow for the case of multiple simultaneous handover transmissions when only local tracks are broadcast. See Figure II-1 for symbology.

9

There are additional advantages to these changes. Initially, we required all nodes detecting the same target to be within broadcast range of each other to keep their common tracks identical. By using the received common track to cancel redundant information, this requirement can be relaxed. Although there are performance advantages to satisfying the requirement, there is no danger of information feedback problems when it is not satisfied.

B. ALGORITHM VALIDATION TESTS

Figure II-5 shows the results of a four-node validation test of the distributed real-time test-bed aeroacoustic tracking algorithms. The test used simulated acoustic measurements corresponding to a low speed target such as a UH-1 helicopter. The figure shows the locations of the nodes and the target position estimates generated by the upper left node. The true track is along a straight line from left to right between the nodes. The nodes have a maximum detection range of 5 kilometers, and broadcast tracks after each new measurement.
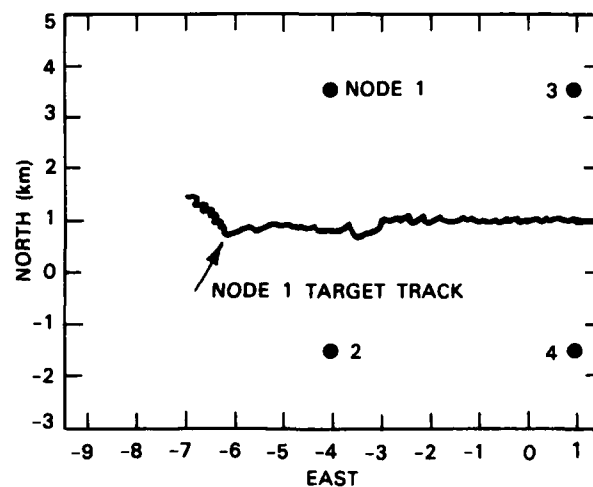
Figure II-5. Results of four-node distributed aeroacoustic tracking algorithm validation test.

The two left-hand nodes cooperate to initiate the track and are the only nodes that track the target until after it has passed between them. The tracking error dies down initially, but increases shortly after the target passes between the first two nodes. In that situation, azimuth measurements from neither node can provide information about the target position along the line joining the two sensors. Due to acoustic delays, this problem manifests itself in the form of position errors only after the target passes between the nodes. Shortly after the target passes between the nodes, there is a sudden decrease in position error, corresponding to the target coming within range of the next two sensors whose measurements provide additional and complementary information. The tracking error stays small out to the point where the the left-hand nodes no longer detect the target and all nodes drop the track.

10

C. TRACKING SOFTWARE MODIFICATIONS

The tracking software is being modified to integrate the tracking algorithm with a video sensor and to exploit the storage capabilities of floppy disk drives attached to each test-bed node. These modifications are straightforward and do not change the way that acoustic measurements are processed. All of the modifications will be completed during the next reporting period.

The TV sensor has a limited field of view and will be controlled to point in the vicinity of targets of interest. The decision of where to look will be made by the TV subsystem itself. The tracking software will provide the TV subsystem with the information it needs to make this decision. In particular, the tracker will send position tracks to the TV sensor subsystem. A software switch, controllable on-line from the User Interface Program, determines whether this information should be sent.

The output from the TV subsystem will consist of target azimuth estimates derived from TV images. These are produced at irregular intervals and may include false measurements. Each measurement, false or true, consists of an estimated target azimuth and a variance for that estimate. The tracking algorithm will accept such measurements, associate them with local position tracks, and update the appropriate track using an Extended Kalman Filter. This Extended Kalman Filter closely resembles that used with acoustic measurements, but is simpler because propagation delay is negligible.

The Nodal Run-Time System (NRTS) currently is being modified to provide a minimal file storage capability using a nodal floppy disk drive. The service more closely resembles a number of small tape cassettes than a traditional file system. The tracking software is being modified to use the NRTS file reading capability to simulate the Sound Processing Subsystem (SPS). It then will be possible to store preprocessed or synthetic acoustic measurement data on a diskette and play it back in a node as if it came over the parallel port interface from the SPS. This will allow us to add nodes to the test-bed that can be used only with such data but which also cost considerably less than a node with an SPS. The tracking software also is being modified to use the NRTS file writing capability to log performance data during experiments. We presently send such information to the VAX-11/780 over an Ethernet. Throughput limitations in the VAX/Ethernet interface do not allow us to record enough of the system performance data in real time. The floppy disk system will circumvent this problem.

D. MULTIHYPOTHESIS TRACKING ALGORITHMS

A joint Lincoln Laboratory and AI&DS effort to develop and evaluate distributed aeroacoustic tracking algorithms based upon a multihypothesis formation and evaluation methodology was initiated during this reporting period. The objective is to determine how their statistical multihypothesis approach can be applied to distributed aeroacoustic tracking and to evaluate possible performance gains and resource requirements trade-offs.

11

The approach is based upon the formation of hypotheses about target tracks, the formation of likelihood functions corresponding to the hypotheses, and the evaluation and pruning of hypotheses on the basis of the likelihood functions. A hypothesis consists of a target track history and a set of measurements associated with the target track. The theory for this approach has been investigated by AI&DS. The joint effort is directed towards a validation of the practical utility of the theory as the basis for distributed tracking algorithms in a DSN.

A development and evaluation plan, extending through FY87, has been formulated and initial tasks completed. The completed tasks include the transfer of Lincoln distributed aeroacoustic tracking algorithms to AI&DS for them to study, transfer of acoustic measurement simulation software to AI&DS, and the preparation of a working paper describing the system context and scenarios of interest.

12

III. AEROACOUSTIC TRACKING ALGORITHM EVALUATION

An evaluation plan was formulated and evaluation activities started for the distributed aeroacoustic aircraft tracking algorithms that have been developed for the DSN test-bed. The goals are to evaluate the Lincoln tracking algorithms, to provide a performance baseline for other algorithms, and to identify knowledge requirements for adaptive algorithm control. The approach is to proceed through a series of evaluation phases from initial algorithm checkout through detailed quantitative performance measurements and final review of the evaluation results. The following summarizes the overall evaluation plan and initial evaluation results.

There are four algorithm evaluation phases: (1) shakedown, (2) tuning, (3) measurement, and (4) knowledge-codification. The phases are distinguished by both purpose and methodology as described below.

In the shakedown phase, the algorithm will be tested using successively more stressing sets of input data and parameter settings. The data and system parameter settings will be varied to uncover algorithm implementation and design bugs. In this phase, the performance evaluation is largely qualitative, ranging from simple confirmations of satisfactory behavior to detailed dissection of the operation of the algorithm. Shakedown evaluation results will be of subsequent use during the knowledge-codification phase that will address how algorithms might be adaptively controlled to improve performance.

The tuning phase will involve experimentation with system parameters to determine robust values for a reasonable range of scenario conditions. It also will reveal which parameters are most critical under different conditions and if there are values that yield significantly better performance for special situations. The algorithm will be applied to many different scenarios for each of several parameter settings. This is different from the shake-down phase in which parameter sets may be applied to only a few scenarios. Experimental results again will be an important input to the knowledge-codification phase.

The measurements phase will gather statistically significant quantitative measures of performance. The shakedown and tuning phases will provide specific scenarios and parameter settings for use in the measurement phase. For each scenario and parameter setting the algorithm will be applied to a number of input data sets that are statistical perturbations for the same scenario. Performance measurements, such as rms tracking error, communications traffic, and average number of false tracks will be collected.

The objective of the knowledge-codification phase will be to review the results of the other three evaluation phases and to identify and document the knowledge requirements for control of tracking algorithms. This will use the results of the other phases and will require no additional experimentation. Most of the information for this phase will be provided by the shakedown and tuning phases, with the measurements phase providing statistical and quantitative confirmation of observations made in the earlier phases.

System shakedown and algorithm tuning experiments have been done with simulated data for a helicopter flying past several nodes, with real UH-1 helicopter data recorded at four nodes and with real jet aircraft data recorded at two nodes.

The multiple node experiments with simulated data confirmed that no undesirable unexpected behavior became evident as the number of tracking and communicating nodes increased from two to six. Sample tracking results are shown in Figure II-5 and discussed in Section II-B.

The real helicopter data was collected using four nodes deployed at Lincoln Laboratory. The experiments with this data confirmed that there is not some critical feature of real data that is not captured by the simulated helicopter data. This is important since it is practical to test algorithms for a much broader range of scenarios with simulated data than with real acoustic data.

The jet aircraft data was collected at a test range in the western United States in cooperation with the Air Vehicle Survivability Evaluation project. The main targets were jet aircraft equipped with a radar beacon transponder that provided true target tracks. The experiments utilized two nodes separated from each other by only 2 km. In one experiment, the target flew between the two nodes and perpendicular to the baseline between them. In the other experiment, the target flew a path approximately tangential to the coverage area, providing a particularly stressing tracking geometry.

The sensitivity of the tracking algorithms to a number of system parameters was investigated using the jet aircraft data. Tracker parameters were varied to observe the sensitivity of the estimation and association algorithms to signal-to-noise ratio thresholds, to parameters controlling data association windows, and to model accelerations and other Kalman filter parameters. Azimuth and position tracks were generated, detection ranges were computed, and tracking accuracies were estimated.

The jet tests involved aircraft operating in the Mach 0.6 to 0.8 speed range. The data analysis confirmed that the wideband array processing algorithm developed under this program will handle high-speed broadband jet targets as well as slow-speed helicopter targets whose spectra are dominated by lines. Signal processing bandpass filters also were varied to observe the effect upon aircraft detection ranges and to remove some narrowband interfering noise sources.

IV. DISTRIBUTED TARGET CLASSIFICATION

In principle, all DSN functions should be implemented in a distributed form. Potential advantages are that communication is minimized and the network will be highly survivable. In this section, we report on research into distributed target classification algorithms.

Distributed classification algorithms will be more difficult to develop than centralized ones. Therefore, it is important to identify some practical circumstances under which there will be substantial benefit to distributed algorithms. One set of circumstances is when the following three conditions hold:

(1) The target is sensed by multiple nodes that provide complementary information.

(2) Information at any single node is insufficient to classify with high confidence.

(3) The amount of raw classification data is so large that it is impractical or inefficient to transmit it to one place for processing.

One situation meeting these conditions may be the acoustic classification of similar targets such as two types of jet aircraft. A situation probably not meeting these conditions would be the discrimination between helicopters and jet fixed wing aircraft.

We have investigated the case of two-node target classification to clarify distributed classification issues. The following summarizes work on two distributed classification approaches. One involves a generalized likelihood ratio test. This approach cannot handle the most general form of likelihood ratio, but it may be adequate in some situations. The second approach focuses on exact distributed implementation of centralized tests. A distributed form of the nearest neighbor rule appears to be feasible for data characterized by arbitrary probability density functions.

Consider two sensors, A and B. Let the sensor A observation vector be x_0 and the sensor B observation vector be y_0. The optimal centralized test that minimizes the probability of error has the form

$$\ln \frac{p_1(x_0, y_0)}{p_2(x_0, y_0)} = \ln \frac{p_1(x_0)}{p_2(x_0)} + \ln \frac{p_1(y_0|x_0)}{p_2(y_0|x_0)} \underset{H_2}{\overset{H_1}{\gtrless}} \ln T \qquad (1)$$

where the subscript i on the probability density function p indicates that the density function is for class i. The notation H_i above and below the inequalities indicates that, for the corresponding sense of the inequality, the statistical hypothesis H_i, 'target belongs to class i,' is selected.

A distributed approximation to this test is obtained if A computes the first term in (1) and B computes an approximation to the second term (the conditional log likelihood ratio) using some estimate for the observations x_0. If B sends the result of this computation to A, then the test (1) can be used to make a decision. A symmetric computation can be made with the roles of A and B reversed. Note that all of this is sensible only if the dimensionality of x_0 and y_0 are high.

15

The decision rule just described has a number of essential differences from the centralized algorithm from which it was derived. First, the performance will be suboptimal compared to the centralized test. Second, if the algorithm is implemented as described as well as with the sensor roles reversed, the two distributed forms can give different results. Some rule is needed to resolve such disagreements. Finally, for certain forms of the probability density functions and certain estimation procedures, a good approximation to the centralized test simply cannot be achieved.

The properties of this class of distributed decision algorithms is dependent on the methods of estimating the unknown observations x_0. If the sensors are allowed to interchange only a single statistic, then the estimate for x_0 must be derived from y_0 and the resulting decision rule is of the form

$$\lambda_A(x_0) + \lambda_B(y_0) \underset{H_2}{\overset{H_1}{\gtrless}} \ln T \quad .$$

This limits how well the distributed test can approximate the centralized test, since in many cases the centralized test will not be separable. If the sensors are permitted to exchange additional statistics, then improved performance is possible.

Table IV-1 lists the algorithm steps when each sensor makes a maximum *a posteriori* probability estimate of the other sensor's observations based on its own. This procedure can be analyzed to compare the decision boundary in the $\underline{x}\,\underline{y}$ combined-observation space of the distributed decision rules to that of the centralized decision rule. In some cases, the decision boundaries are quite close. However, in general, the absence of cross products between $\underline{x}$ and $\underline{y}$ limits the closeness of the approximation.

An alternative approach to distributed classification is to try to implement exactly a centralized test in a distributed manner. This is certainly possible if the log likelihood ratio (or more generally, the discriminant function for the classifier) is *separable* i.e.,

$$\lambda(x_0, y_0) = f(x_0) + g(y_0) \quad .$$

This will happen for the special cases of Gaussian measurements with equal class covariances or in the case of a linear discriminant function designed by the Fischer criterion.* It also clearly occurs when the measurements x_0 and y_0 are independent.

More generally, if the discriminant function can be expressed as a sum or product of a small number of statistics, each a function of only x_0 or y_0, then an efficient distributed decision rule will result. One such distributed procedure might be developed for the nearest neighbor rule.* The nearest neighbor rule is a nonparametric approach to probability density estimation and

---

* K.Fukunaga, *Introduction to Statistical Pattern Recognition* (Academic Press, New York, 1972).

<table>
<tr><th colspan="2">TABLE IV-1

Algorithm Steps</th></tr>
</table>

SENSOR A	SENSOR B
1. Get observations $\underline{x}_0$	1. Get observations $\underline{y}_0$
2. Compute $\lambda_A = \ln \dfrac{p_1(\underline{x}_0)}{p_2(\underline{x}_0)}$	2. Compute $\lambda_B = \ln \dfrac{p_1(\underline{y}_0)}{p_2(\underline{y}_0)}$
3. Estimate $\hat{\underline{y}}_1 = \max_y\ p_1(\underline{y}\mid\underline{x}_0)$ $\hat{\underline{y}}_2 = \max_y\ p_2(\underline{y}\mid\underline{x}_0)$	3. Estimate $\hat{\underline{x}}_1 = \max_x\ p_1(\underline{x}\mid\underline{y}_0)$ $\hat{\underline{x}}_2 = \max_x\ p_2(\underline{x}\mid\underline{y}_0)$
4. Compute $\lambda'_A = \ln \dfrac{p_1(\underline{x}_0\mid\hat{\underline{y}}_1)}{p_2(\underline{x}_0\mid\hat{\underline{y}}_2)}$	4. Compute $\lambda'_B = \ln \dfrac{p_1(\underline{y}_0\mid\hat{\underline{x}}_1)}{p_2(\underline{y}_0\mid\hat{\underline{x}}_2)}$
5. Receive λ'_B from B	5. Receive λ'_A from A
6. Compute and compare: $\lambda_A + \lambda'_B \mathop{\lessgtr}\limits_{H_2}^{H_1} \ln T$	6. Compute and compare: $\lambda'_A + \lambda_B \mathop{\lessgtr}\limits_{H_2}^{H_1} \ln T$

leads to a direct implementation of the likelihood ratio. It involves distance computations that are separable in $\underline{x}$ and $\underline{y}$, therefore it is a likely candidate for distributed implementation. A first cut distributed implementation neighbor rule could proceed as follows. Measurement data for different target classes are obtained and represented as points in the $\underline{x}\,\underline{y}$ measurements space. New measurements $\underline{x}_0$, $\underline{y}_0$ on a target to be classified then would be processed as follows. Node A would compute distances to the p nearest points in the x subspace and send these to B. Node B would compute distances to the p nearest points in y subspace and send these to A. Both nodes then would select the point whose distance in the x y space was closest to $(x_0\ y_0)$ as defining the class of the target. In cases where the two sets of p points had no intersection, the process would have to iterate with a larger set of points.

V. COOPERATIVE USE OF ACOUSTIC AND TV SENSORS

A test-bed TV subsystem is being developed to demonstrate the cooperative use of sensors with complementary properties and to investigate the management of the interaction between these sensors. An immediate objective is to demonstrate acoustic cueing of the TV subsystem. The tracker will send a cueing message to the TV subsystem every few seconds. Upon receiving the message, which may include information on multiple targets, the TV subsystem will select a target, point the camera toward it, process the resulting images, measure the target azimuth, and return the measurement to the tracking system. Recent work on the video subsystem development is summarized in this section. Work on the tracking system to provide cues and process TV azimuth measurements was described in Section II-C.

Actual system development was preceded by an analysis and algorithm development effort that identified four modules required for the TV subsystem. The four modules are: (1) track selection, (2) camera positioning, (3) image processing, and (4) target detection. Preliminary versions of the algorithms for each of these modules then were developed.

The track-selection module processes target position and velocity lists provided by the tracking system and selects a target for TV tracking. Target selection must take many factors into account to maximize the probability that a useful TV azimuth measurement will be obtained. Figure V-1 identifies several of these factors. For example, the track selection module will reject targets for which the camera slew is insufficient to keep up with the azimuth rate of the target. As another example, the system will reject targets that are too distant to be detected by the video system. Preliminary selection algorithms have been developed that treat several of these factors.
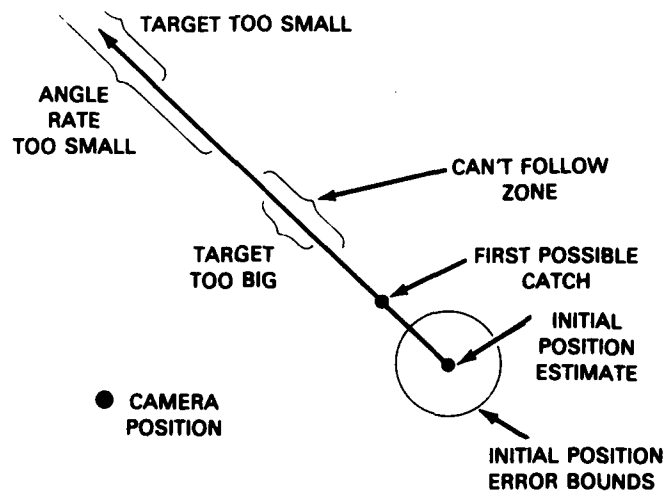

Figure V-1. Target selection factors for the TV subsystem.

19

The camera-positioning module takes the present camera azimuth and the present position and velocity of a selected target as input and generates commands to point the camera for target acquisition. An approximate algorithm to do this has been developed and demonstrated to perform satisfactorily for targets that have small azimuth rates of change.

The image-processing module takes as input the pixel values corresponding to one or more TV frames and produces an image suitable for the determination of target azimuth. Two algorithms were considered: a single-frame target-contrast algorithm and a two-frame frame-differencing algorithm. The performance and computational efficiency of these two algorithms was evaluated and the frame-differencing algorithm was selected for use.

The target-detection module takes the image produced by the image-processing module (e.g., a difference image) and produces an azimuth measurement. An algorithm to achieve this objective was designed using the techniques of column summation and elevation hopping illustrated in Figure V-2. Column summation is used to generate a histogram of image-column brightness. The peak of this histogram identifies the measured target azimuth. Elevation hopping is used to scan two adjacent horizontal strips of the image. If the target is found in the low elevation strip, the higher elevation strip is not searched.
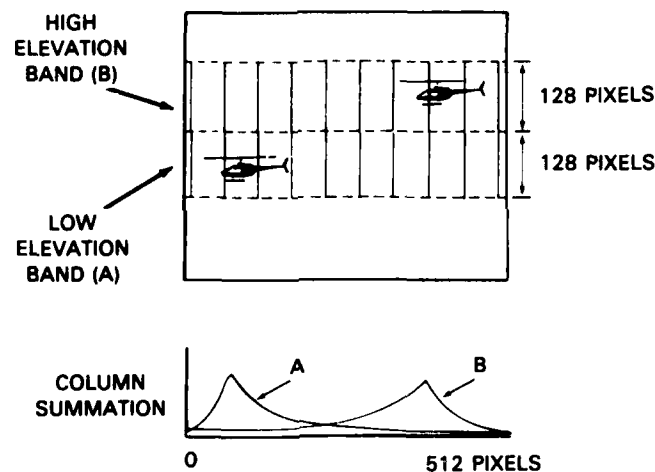

Figure V-2. Target detection algorithm based upon column summation of frame differences over two target elevation bands.

Real-time versions of the four TV subsystem modules now are being developed and integrated. A simple camera-positioning module has been developed and tested. It employs an approximate method to predict target and camera azimuths and uses these predictions to generate camera commands. Previously developed image-processing software is being integrated with the other modules. Software also was developed to implement the column sum and elevation hopping features of the target-detection module. Other minor changes to the existing software were made to achieve improved test capabilities and performance.

20

Initial evaluation of the TV subsystem hardware identified the need for an improved mount and camera position measurement capability. A new mount is on order that will increase the camera slew rate by a factor of two to twelve degrees per second, reduce backlash from a few degrees to a fraction of a degree and is constructed such that it will be easy to add hardware to obtain precise measurements of the camera azimuth. Modifications are being designed for the camera position measurement system to reduce measurement errors to less than a tenth of a degree.

VI. KNOWLEDGE-BASED SYSTEM DIAGNOSIS

During this report period, we have made significant progress towards implementing an automated DSN system diagnosis capability. The implementation is being carried out on a Symbolics 3600 Lisp Machine. In contrast to our previous rule-based system, the current system utilizes some of the latest A.I. techniques such as qualitative reasoning and hierarchical planning.

The system will take as input the data produced by a DSN system as well as a description of 'expectations' concerning target positions, tracks, and types. If there is a discrepancy between the DSN system data and expectations, the diagnosis system will produce an explanation for the discrepancy. The explanations will be similar in nature to those given by an expert who is familiar with the underlying theory for the DSN tracking and signal processing algorithms. Theoretical models underlying the DSN algorithms are being incorporated into the diagnosis system to achieve this. A significant portion of this knowledge is organized in the form of a 'search-space' and an associated set of 'operators'. The search-space is organized in the form of abstract 'states' that parametrically represent the possible outputs of the DSN system. The operators represent various actions of the DSN system that transform one state into another.

Each state is represented as a collection of 'signal-objects' corresponding to the acoustic sources at a particular time. Each of the signal-objects is characterized further in terms of its spectral characteristics (e.g. bandwidth, amplitude, energy) as well as dynamic characteristics (e.g., velocity, track history). Each state is implemented so that it can be described at multiple levels of abstraction; the highest level of abstraction hides the most detail. The current implementation handles four levels of abstraction. The parameters of each state are specified as a range of values rather than a fixed known value. Thus, for example, an aircraft whose direction is unknown can be represented as having a direction in the range from 0 to 360 degrees.

Each operator is implemented as a Lisp-function that manipulates the data structures representing the states. Since state parameters are specified as ranges of values, the operators manipulate parameters that are specified as ranges. Furthermore, an operator can carry out actions at any level of abstraction. In addition to the Lisp-function that carries out the action of an operator on a state, each operator also consists of a symbolic description of the conditions under which it can be applied to a state. So far, the major operators required for representing the DSN signal processing system have been implemented.

The diagnosis system forms an explanation by performing means-ends analysis[1] to select operators that will transform a state representing the expected track-data into a state representing the measured track data. The collection of operators and the intermediate data states generated in this process constitute the explanation. A user-interface that diagramatically illustrates the explanation for the user has been designed and partially implemented.

We also have continued to refine the design of other portions of the diagnosis system. In particular, we have concentrated our efforts on the subsystem that takes as input the explanation produced from the means-ends analysis and, using qualitative reasoning,[2,3] produces a suggestion for how to change the system parameters to improve the DSN output.

REFERENCES

1. A. Newell and H.A. Simon, *Human Problem Solving*, Prentice Hall, Englewood Cliffs, New Jersey, 1972.

2. B.J. Kuipers, Commonsense Reasoning About Causality: Deriving Behavior from Structure, Artificial Intelligence, **24**, No. 1, (1984).

3. K.D. Forbus, Qualitative Process Theory, MIT Artificial Intelligence Laboratory Memo 664 (1982).

VII. OTHER TEST-BED ACTIVITIES

The transition of research and software development activities from a version 7 Unix operating on a PDP-11/70 to a Version 4.2 Berkeley Unix operating on a VAX-11/780 was completed during this reporting period. Target tracking, test-bed simulation and display software all were converted to the new system. A floppy disk driver was written for the VAX to support the creation of disks for booting test-bed nodes. Nodal software support including an MC68000 cross-compiler and link editor were converted for use on the VAX. The User Interface Program (UIP), which allows users to control test-bed experiments by communicating with the nodes over an Ethernet, was modified for the new VAX system.

Subsequent to the VAX conversion, real-time tests of the test-bed system uncovered a number of minor problems that were corrected. The major one of these was that clocks in different test-bed nodes drifted out of synchronism by large and variable amounts, depending upon system load. A bug related to the priority of clock functions was uncovered in the nodal run time system and corrected.

Plans call for the future use of multiple user workstations to access and control the test-bed system. Considerable progress has been made towards implementing user interface functions on Silicon Graphics workstations that will be used for that purpose. The Silicon Graphic workstations are MC68000 Unix workstations with extensive color graphics options. A version of the UIP was installed on such a workstation. Some additional work is required to make use of the color graphics and to provide control lines to remotely restart and reboot the nodes.

The DSN test-bed has been used to run experiments with up to six nodes and is in use on an almost daily basis. Some system reliability problems were experienced initially but are now under control. The problems were brought under control by instituting routine test and maintenance schedules and by implementing and using a number of software tools to help track problems. The tools are VAX programs used to record the results of all test-bed runs and to maintain a cross referenced list of problems. These have been helpful for identifying hardware failure patterns and for indentifying failures resulting from operator error. The occurrence of operator errors suggested changes to be made in the user interface program. Most of the changes have been to automate procedures and reduce the amount of interactive inputs and responses required of the user.

The basic DSN test-bed includes six nodes, each complete with sensors, signal processing subsystem, and standard nodal computers that are used for tracking and communication. Three additional nodes have been built and are being integrated into the test-bed to allow experiments with up to nine nodes using synthetic data. The additional nodes will be restricted to using synthetic or preprocessed acoustic data because they contain only standard nodal computers. The sensor and signal processing system will be simulated by reading from small disks attached to the nodes. The additional nodes also can serve as spares.

25

DSN radio broadcast software has been implemented and integrated into the nodal run time system and now is undergoing tests before integration with actual radios and with the tracking application. Additional software also was developed to support radio hardware testing. This included the radio diagnostic program to make use of the 8089 I/O processor that is a component of the radio interface board.

A small amount of multiple node tracking data has been collected to begin the investigation of multisite data integration algorithms.

GLOSSARY

AI&DS   Advanced Information and Decision Systems

DSN   Distributed Sensor Networks

MACSYMA   A symbolic mathematics manipulation program

NRTS   Nodal Run-Time System

SATS   Semiannual Technical Summary

SPS   Sound Processing Subsystem

UIP   User Interface Program

27

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-85-237	2. GOVT ACCESSION NO. *AD-A160596*	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Distributed Sensor Networks		5. TYPE OF REPORT & PERIOD COVERED Semiannual Technical Summary 1 October 1984 — 31 March 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Richard T. Lacoss		8. CONTRACT OR GRANT NUMBER(s) F19628-85-C-0002
9. PERFORMING ORGANIZATION NAME AND ADDRESS Lincoln Laboratory, MIT P.O. Box 73 Lexington, MA 02173-0073		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Element Nos.62708E Project Nos.5D30 & 5T10 ARPA Order 3345
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209		12. REPORT DATE ~~31 March 1985~~ *26 SEPT. 85*
		13. NUMBER OF PAGES 38
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* Electronic Systems Division Hanscom AFB, MA 01731		15. SECURITY CLASS. *(of this Report)* Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT *(of this Report)* Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*		
18. SUPPLEMENTARY NOTES None		

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

multiple-sensor surveillance system	acoustic sensors	digital radio
distributed tracking	video sensors	distributed estimation
target surveillance and tracking	low-flying aircraft	knowledge-based data interpretation
communication network	acoustic array processing	

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report describes the work performed on the DARPA Distributed Sensor Networks Program at Lincoln Laboratory during the period 1 October 1984 through 31 March 1985.